# Safe Agents in Space: Lessons from the Autonomous Sciencecraft Experiment

Rob Sherwood, Steve Chien, Daniel Tran, Benjamin Cichy, Rebecca Castano, Ashley Davies, Gregg Rabideau

Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Dr., Pasadena, CA 91109
{firstname.lastname@jpl.nasa.gov}

**Abstract.** An Autonomous Science Agent is currently flying onboard the Earth Observing One Spacecraft. This software enables the spacecraft to autonomously detect and respond to science events occurring on the Earth. The package includes software systems that perform science data analysis, deliberative planning, and run-time robust execution. Because of the deployment to a remote spacecraft, this Autonomous Science Agent has stringent constraints of autonomy, reliability, and limited computing resources. We describe these constraints and how they are reflected in our agent architecture.

## 1. Introduction

The Autonomous Sciencecraft Experiment (ASE) is currently flying autonomous agent software on the Earth Observing One (EO-1) spacecraft [**Error! Reference source not found.**]. This software demonstrates several integrated autonomy technologies to enable autonomous science. Several algorithms are used to analyze remote sensing imagery onboard in order to detect the occurrence of science events. These algorithms will be used to downlink science data only on change, and will detect features of scientific interest such as volcanic eruptions, flooding, ice breakup, and presence of cloud cover. The results of these onboard science algorithms are inputs to onboard decision-making algorithms that then modifies the spacecraft observation plan to capture high value science events. This new observation plan is then be executed by a robust goal and task oriented execution system, able to adjust the plan to succeed despite run-time anomalies and uncertainties. Together these technologies enable autonomous goal-directed exploration and data acquisition to maximize science return. This paper describes the Autonomous Sciencecraft Experiment (ASE) effort to develop and deploy the Autonomous Science Agent on the Earth Observing One spacecraft.

The ASE onboard flight software includes several autonomy software components:

- Onboard science algorithms that will analyze the image data to detect trigger conditions such as science events, "interesting" features, changes relative to previous observations, and cloud detection for onboard image masking

- Robust execution management software using the Spacecraft Command Language (SCL) [7] package to enable event-driven processing and low-level autonomy
- The Continuous Activity Scheduling Planning Execution and Replanning (CASPER) [3] software that will replan activities, including downlink, based on science observations in the previous orbit cycles

The onboard science algorithms will analyze the images to extract static features and detect changes relative to previous observations. Prototype software has already been demonstrated on EO-1 Hyperion data to automatically identify regions of interest including land, ice, snow, water, and thermally hot areas. Repeat imagery using these algorithms can detect regions of change (such as flooding and ice melt) as well as regions of activity (such as lava flows). Using these algorithms onboard will enable retargeting and search, e.g., retargeting the instrument on a subsequent orbit cycle to identify and capture the full extent of a flood. On future interplanetary space missions, onboard science analysis will enable capture of short-lived science phenomena. These can be captured at the finest time-scales without overwhelming onboard memory or downlink capacities by varying the data collection rate on the fly. Examples include: eruption of volcanoes on Io, formation of jets on comets, and phase transitions in ring systems. Generation of derived science products (e.g., boundary descriptions, catalogs) and change-based triggering will also reduce data volumes to a manageable level for extended duration missions that study long-term phenomena such as atmospheric changes at Jupiter and flexing and cracking of the ice crust and resurfacing on Europa.

The planning software (CASPER) will generate mission operations plans from goals provided by the onboard science analysis module. The model-based planning algorithms will enable rapid response to a wide range of operations scenarios based on a deep model of spacecraft constraints, including faster recovery from spacecraft anomalies. The onboard planner will accept as inputs the science and engineering goals and ensure high-level goal-oriented behavior.

The robust execution system (SCL) accepts the CASPER-derived plan as an input and expands the plan into low-level commands. SCL monitors the execution of the plan and has the flexibility and knowledge to perform event-driven commanding to enable local improvements in execution as well as local responses to anomalies.

A typical ASE scenario involves monitoring of active volcano regions such as Mt. Etna in Italy. (See Figure 1.) Hyperion data have been used in ground-based analysis to study this phenomenon. The ASE concept will be applied as follows:

1. Initially, ASE has a list of science targets to monitor that have been sent as high-level goals from the ground.
2. As part of normal operations, CASPER generates a plan to monitor the targets on this list by periodically imaging them with the Hyperion instrument. For volcanic studies, the infra-red and near infra-red bands are used.
3. During execution of this plan, the EO-1 spacecraft images Mt. Etna with the Hyperion instrument.
4. The onboard science algorithms analyze the image and detect a fresh lava flow, or active vent. If new activity is detected, a science goal is generated to continue

monitoring the volcanic site. If no activity is observed, the image is not downlinked.

5. Assuming a new goal is generated, CASPER plans to acquire a further image of the ongoing volcanic activity.
6. The SCL software executes the CASPER generated plan to re-image the site.
7. This cycle is then repeated on subsequent observations.

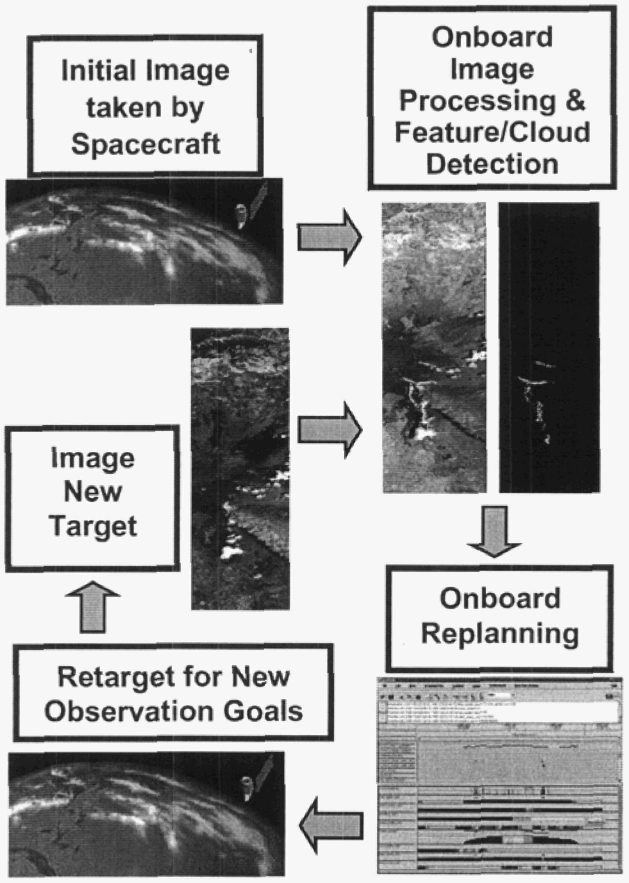
Initial Image taken by Spacecraft
Onboard Image Processing & Feature/Cloud Detection
Image New Target
Onboard Replanning
Retarget for New Observation Goals

**Fig. 1.** Autonomous Science Scenario

Building autonomy software for space missions has a number of key challenges; many of these issues increase the importance of building a reliable, safe, agent.

1. Limited, intermittent communications to the agent. A typical spacecraft in low earth orbit (such as EO-1) has eight 10-minute communications opportunities per day. This means that the spacecraft must be able to operate for long periods of time without supervision. For deep space missions the spacecraft may be in communications far less frequently. Some deep space missions only contact the spacecraft once per week, or even once every several weeks.

2. Spacecraft are very complex. A typical spacecraft has thousands of components, each of which must be carefully engineered to survive rigors of space (extreme temperature, radiation, physical stresses). Add to this the fact that many components are one-of-a-kind and thus have behaviors that are hard to characterize.
3. Limited observability. Because processing telemetry is expensive, onboard storage is limited, and downlink bandwidth is limited, engineering telemetry is limited. Thus onboard software must be able to make decisions based on limited information and ground operations teams must be able to operate the spacecraft with even more limited information.
4. Limited computing power. Because of limited power onboard, spacecraft computing resources are usually very constrained. On average, spacecraft CPUs offer 25 MIPS and 128 MB RAM – far less than a typical personal computer. Our CPU allocation for ASE on EO-1 is 4 MIPS and 128MB RAM.
5. High stakes. A typical space mission costs hundreds of millions of dollars, any failure has significant economic impact. The total EO-1 Mission cost is over $100 million. Planetary missions can have severe launch constraints due to planetary geometries. In these cases, if a space mission is lost it may be years before another similar mission can be launched. Additionally, space missions can take years to plan, build, launch, and reach their targets. This delay can be catastrophic.

Of the above aspects of spacecraft autonomy, two critical issues are:

1. Extreme reliability – because of the extreme cost of space missions, and inability to access the spacecraft except by communications, the software agent must be exceptionally reliable.
2. CPU and RAM performance – spacecraft have extremely limited CPU and RAM (in our case 4 MIPS and 128MB RAM) yet must adhere to at least soft, real-time constraints.

In the remainder of this paper we describe the ASE software architecture and components. We then discuss how the issues of reliability and performance affected the software architecture.

## 2. The EO-1 Mission

EO-1 is the first satellite in NASA's New Millennium Program Earth Observing series. The primary focus of EO-1 is to develop and test a set of advanced technology land imaging instruments.

EO-1 was launched on a Delta 7320 from Vandenberg Air Force Base on November 21, 2000. It was inserted into a 705 km circular, sun-synchronous orbit at a 98.7 degrees inclination. This orbit allows for 16-day repeat tracks, with 3 over flights per 16-day cycle with a less than 10-degree change in viewing angle. For each scene, over 20-Gbits of data from the Advanced Land Imager (ALI), Hyperion, and Atmospheric Corrector (AC) are collected and stored on the onboard solid-state data recorder at high rates.

The ASE described in this paper uses the Hyperion hyper-spectral instrument. The Hyperion is a high-resolution imager capable of resolving 220 spectral bands (from 0.4 to 2.5 μm) with a 30-meter spatial resolution. The instrument images a 7.5 km by 42 km land area per image and provides detailed spectral mapping across all 220 channels with high radiometric accuracy.

The EO-1 spacecraft has two Mongoose M5 processors. The first M5 is used for the EO-1 command and data handling functions. The other M5 is part of the WARP (Wideband Advanced Recorder Processor), a large mass storage device. Each M5 runs at 12 MHz (for ~8 MIPS) and has 256 MB RAM. Both M5's run the VxWorks operating system. The ASE software operates on the WARP M5. This provides an added level of safety for the spacecraft since the ASE software does not run on the main spacecraft processor.

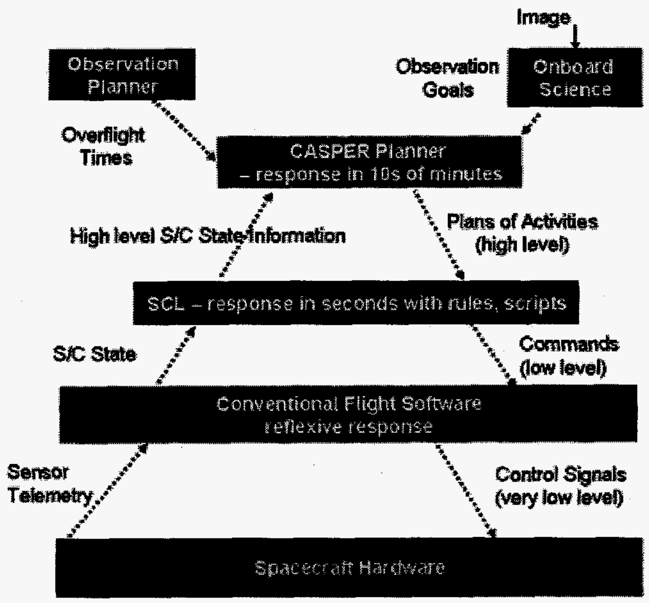
Image
Observation Planner
Observation Goals
Onboard Science
Overflight Times
CASPER Planner – response in 10s of minutes
High level S/C State Information
Plans of Activities (high level)
SCL – response in seconds with rules, scripts
S/C State
Commands (low level)
Conventional Flight Software reflexive response
Sensor Telemetry
Control Signals (very low level)
Spacecraft Hardware

**Fig. 2.** Autonomy Software Architecture

## 3. Autonomy Software Architecture

The autonomy software on EO-1 is organized into a traditional three-layer architecture [5] (See Figure 2.). At the highest level of abstraction, the Continuous Activity Scheduling Planning Execution and Replanning (CASPER) software is responsible for mission planning functions. CASPER schedules science activities while respecting spacecraft operations and resource constraints. The duration of the planning process is on the order of tens of minutes. CASPER scheduled activities are inputs to the Spacecraft Command Language (SCL) system, which generates the detailed sequence commands corresponding to CASPER scheduled activities. SCL operates on the several second timescale. Below SCL, the EO-1 flight software is responsible for lower level control of the spacecraft and also operates a full layer of independent fault protection. The interface from SCL to the EO-1 flight software is at the same level as ground generated command sequences. The science analysis software is scheduled by CASPER and executed by SCL in a batch mode. The results from the science analysis software result in new observation requests presented to the CASPER system for integration in the mission plan.

This layered architecture was chosen for two principal reasons:

1. The layered architecture enables separation of responses based on timescale and most appropriate representation. The flight software level must implement control loops and fault protection and respond very rapidly and is thus directly coded in C. SCL must respond quickly (in seconds) and perform many procedural actions. Hence SCL uses as its core representation scripts, rules, and database records. CASPER must reason about longer term operations, state, and resource constraints. Because of its time latency, it can afford to use a mostly declarative artificial intelligence planner/scheduler representation.
2. The layered architecture enables redundant implementation of critical functions – most notable spacecraft safety constraint checking. In the design of our spacecraft agent model, we implemented spacecraft safety constraints in all levels where feasible.

Each of the software modules operates at a separate VxWorks priority. The tasks are shown below in Table 1 in decreasing priority. The ASE to FSW bridge is the task responsible for reading the real-time flight software telemetry stream, extracting pertinent data, and making it accessible to the remainder of the ASE software. The Band Stripping task reads the science data from the onboard WARP solid state recorder and extracts a small portion of the science data (12 bands of Hyperion data) to RAM. The science analysis software then operates on the extracted data to detect science events.

It is worth noting that our agent architecture is designed to scale to multiple agents. Agents communicate at either the planner level (via goals) or the execution level (to coordinate execution).

**Table 1.** EO-1 Software Tasks in Decreasing Task Priority (e.g. upper tasks have highest priority for CPU).

| Set of Tasks | Rationale for Priority |
|---|---|
| EO-1 Flight Software | Required for WARP hardware safety |
| ASE to FSW Bridge | Required to keep up with telemetry stream |
| Band Stripping | Utilizes WARP hardware while running |
| SCL | Lowest level autonomy, closes tightest loops |
| CASPER | Responds in tens of minutes timescale |
| Science Analysis | Batch process without hard deadlines |

We now describe each of the architectural components of our architecture in further detail.

## 4. Onboard Science Analysis

The first step in the autonomous science decision cycle is detection of interesting science events. In the complete experiment, a number of science analysis algorithms will be flown including:

- Thermal anomaly detection – uses infrared spectra peaks to detect lava flows and other volcanic activity.
- Cloud detection [13] – uses intensities at six different spectra and thresholds to identify likely clouds in scenes.
- Flood scene classification – uses ratios at several spectra to identify signatures of water inundation as well as vegetation changes caused by flooding.
- Change detection – uses multiple spectra to identify regions changed from one image to another. This technique is applicable to many science phenomena including lava flows, flooding, freezing and thawing and is used in conjunction with cloud detection.
- Generalized Feature detection – uses trainable recognizers to detect spatial features as sand dunes and wind streaks.

All of these science algorithms use the Hyperion instrument, as the ALI data is not available for processing onboard. These algorithms are described in more detail in [4].

## 5. Onboard Mission Planning

In order for the spacecraft to respond autonomously to a science event, it must be able to independently perform the mission planning function. This requires software that can model all spacecraft and mission constraints. The CASPER [3] software performs this function for ASE. CASPER represents the operations constraints in a general modeling language and reasons about these constraints to generate new operations plans that respect spacecraft and mission constraints and resources. CASPER uses a local search approach [12] to develop operations plans.

Because onboard computing resources are scarce, CASPER must be very efficient in generating plans. While a typical desktop or laptop PC may have 2000-3000 MIPS performance, 5-20 MIPS is more typical onboard a spacecraft. In the case of EO-1, the Mongoose V CPU has approximately 8 MIPS. Of the three software packages, CASPER is by far the most computationally intensive. For that reason, our optimization efforts were focused on CASPER. Careful engineering and modeling were required to enable CASPER to build a plan in tens of minutes on the relatively slow CPU.

CASPER is responsible for long-term mission planning in response to both science goals derived onboard as well as anomalies. In this role, CASPER must plan and schedule activities to achieve science and engineering goals while respecting resource and other spacecraft operations constraints. For example, when acquiring an initial image, a volcanic event is detected. This event may warrant a high priority request for a subsequent image of the target to study the evolving phenomena. In this case, CASPER will modify the operations plan to include the necessary activities to re-image. This may include determining the next over flight opportunity, ensuring that the spacecraft is pointed appropriately, that sufficient power and data storage are available, that appropriate calibration images are acquired, and that the instrument is properly prepared for the data acquisition.

In the context of ASE, CASPER reasons about the majority of spacecraft operations constraints directly in its modeling language. However, there are a few notable exceptions. First, the over flight constraints are calculated using ground-based orbit analysis tools. The over flight opportunities and pointing required for all targets of interest are uploaded as a table and utilized by CASPER to plan. Second, the ground operations team will initially perform management of the momentum of the reaction wheels for the EO-1 spacecraft. This is because of the complexity of the momentum management process caused by the EO-1 configuration of three reaction wheels rather than four.

## 6. Onboard Robust Execution

ASE uses the Spacecraft Command Language (SCL) [7] to provide robust execution. SCL is a software package that integrates procedural programming with a real-time, forward-chaining, rule-based system. A publish/subscribe software bus allows the distribution of notification and request messages to integrate SCL with other onboard software. This design enables both loose or tight coupling between SCL and other flight software as appropriate.

The SCL "smart" executive supports the command and control function. Users can define scripts in an English-like manner. Compiled on the ground, those scripts can be dynamically loaded onboard and executed at an absolute or relative time. Ground-based absolute time script scheduling is equivalent to the traditional procedural approach to spacecraft operations based on time. In the EO-1 experiment, SCL scripts will also be planned and scheduled by the CASPER onboard planner. The science analysis algorithms and SCL work in a cooperative manner to generate new goals for CASPER. These goals are sent as messages on the software bus.

Many aspects of autonomy are implemented in SCL. For example, SCL implements many constraint checks that are redundant with those in the EO-1 fault protection software. Before SCL sends each command to the EO-1 command processor, it undergoes a series of constraint checks to ensure that it is a valid command. Any pre-requisite states required by the command are checked (such as the communications system being in the correct mode to accept a command). SCL will also verify that there is sufficient power so that the command does not trigger a low bus voltage condition and that there is sufficient energy in the battery. Using SCL to check these constraints (while included in the CASPER model) provides an additional level of safety to the autonomy flight software.

## 7. Agent Safety requirements

Because of significant concerns for spacecraft health, ASE implements a layered redundant approach to enforcing spacecraft safety. This means that whenever possible *at every level* of the agent architecture, redundant checks are implemented to enhance spacecraft safety. This level of redundancy is enhanced by the fact that each of the software levels in the ASE architecture is implemented by a separate set of VxWorks tasks.

Each level of software in the ASE architecture is designed to operate safely in the presence of a malfunction from the higher level of ASE software. For example, if the ASE science software goes haywire and requests 30 observations in a single orbit, the CASPER planner is designed to reject the unschedulable goals and only schedule a single observation for that orbit. If the CASPER planner should schedule overlapping observations SCL is designed to reject the contradictory commands. Or if CASPER should plan to acquire data, but omit key setup steps, SCL is designed to abort the observation. Likewise, if SCL sends an improper instrument setup sequence, the EO-1 flight software is designed to reject commands that would endanger the spacecraft.

The EO-1 spacecraft engineers, EO-1 operations personnel, as well as ASE team members have reviewed these safeguards (for a more detailed description of the model development, validation, and testing process, see [13]). In addition, automated code generation techniques were used to develop SCL state & resource constraint checks directly from the CASPER model.

Table 2 shows the redundant safeguards implemented in the ASE software, EO-1 flight software, and ASE and EO-1 operations procedures as relating to two spacecraft safety constraints. As shown, the operations team, the CASPER planner (via its model), SCL (via scripts and rules), and the EO-1 flight software each implement constraints to protect the spacecraft from damage due to faulty commands or anomalies. In this manner, even if one of the layers malfunctions, the spacecraft may still be protected.

**Table 2.** Redundant Safety Contraints implemented for two safety concerns

| Implement level | Instruments overheat from being left on too long | Instruments exposed to sun |
|---|---|---|
| Operations Team | For each turn on command, look for the following turn off command. Verify that they are within the maximum separation. | Verify orientation of spacecraft during periods when instrument covers are open. |
| CASPER | High-level activity decomposes into turn on and turn off activities that are with the maximum separation. | Maneuvers must be planned at times when the covers are closed (otherwise, instruments are pointing at the earth) |
| SCL | Rules monitor the "on" time and issue a turn off command if left on too long. | Constraints prevent maneuver scripts from executing if covers are open. |
| EO-1 Flight Software | Fault protection software will shut down the instrument if left on too long. | Fault protection will safe the spacecraft if covers are open and pointing near the sun. |

One major exception to the design guideline of maximum separation and redundancy of software elements is that the WARP software and ASE software shares the same memory space. Because of this situation, a pointer error or over-run of a buffer in the ASE software could cause a software exception in the WARP flight software (or vice versa). Likewise a memory error in one ASE component could cause a software exception in another ASE component. All software exceptions on the WARP M5 processor are handled by a software reset of the WARP M5. Thus any such anomaly in any of the software is likely to cause such a reset. Unfortunately, the version of the VxWorks being flown on EO-1 (5.3.1) does not support process (task) based memory spaces. VxWorks 6.0 and beyond are expected to support this feature. However, the most commonly used version of VxWorks for Space applications is 5.4.1, so this enhancement may not become prevalent in the near term.

## 8. Performance Issues in ASE

The ASE experiment is constrained by the computing environment onboard EO-1. Because the EO-1 software builds are each a single static image, all ASE components that dynamically allocated RAM required flight of their own memory manager. Originally SCL flew with a legacy memory manager previously used with SCL on the FUSE mission. CASPER used a separate memory manager adapted from JPL's Deep Impact mission. However, performance from early flight tests indicated that the SCL memory manager was significantly hampering performance, so SCL was switched to use the same memory manager as CASPER (but with its own heap space). Note that these memory managers had to not only allocate and de-allocate memory quickly but also not suffer from longer term issues such as fragmentation.

The VxWorks priorities of the ASE software were determined by design guidelines but were then analyzed based on performance data. The very limited CPU onboard meant that long duration scenario tests in successive ground testbeds followed by incremental flight tests were needed to ensure scaleability.

In addition, both SCL and CASPER required that modeling and operations be influenced by limited onboard computing and response needs. For example, initially within SCL a much larger set of safety constraints was modeled and execution was designed to be much more closed loop. However, testbed runs and early flight tests indicated that telemetry delays and CPU bottlenecks meant that this design was delaying time-sensitive commands. Most importantly, instrument on-times were delayed (e.g. late) and too long (resulting in extra data acquired). The ASE team was forced to both streamline the code (including the memory manager modification) and streamline the model to speed execution.

The CASPER planner is also a significant user of onboard CPU. When CASPER is planning future observations it utilizes all of the available CPU and takes approximately 8 minutes to plan each observation. The CASPER model was designed to operate within a minimal CPU profile – and as a result observations are planned with less flexibility. By utilizing more decompositions instead of subgoals and by fixing temporal offsets rather than retaining flexibility, search is reduced and response time improved at the cost of plan quality (in some cases).

## 9. Flight Status

The ASE software has steadily progressed to full operations with the major milestones listed below. We have begun full operations with flight of the integrated science with autonomous planning and execution and are steadily increasing the tempo of operations. This software will continue to be flown until at least December 2004 and will be used to acquire as many science-triggered scenes as resources allow.

| Test Description | Test Date |
|---|---|
| Onboard cloud detection test | March 2003 |
| Onboard commanding path test | May 2003 |
| CASPER ground generated commands executed onboard | July 2003 |
| Software jumping and loading test | August 2003 |
| ASE autonomously acquires dark calibration image and performs downlink | October 2003 |
| ASE autonomously acquires science images and performs downlinks | Jan. 2004 – Feb. 2004 |
| ASE autonomously analyzes science data onboard and triggers subsequent observations | Apr. 2004 - Dec. 2004 |

## 10. Related work and Summary

In 1999, the Remote Agent experiment (RAX) [10] executed for a few days onboard the NASA Deep Space One mission. RAX is an example of a classic three-tiered architecture [6], as is ASE. RAX demonstrated a batch onboard planning capability (as opposed to CASPER's continuous planning) and RAX did not demonstrate onboard science. PROBA [11] is a European Space Agency (ESA) mission demonstrates onboard autonomy and launched in 2001. However, ASE has more of a focus on model-based autonomy than PROBA.

The Three Corner Sat (3CS) University Nanosat mission will be using the CASPER onboard planning software integrated with the SCL ground and flight execution software [1]. The 3CS mission is scheduled for launch on a Delta IV rocket July 3, 2004. The 3CS autonomy software includes onboard science data validation, replanning, robust execution, and multiple model-based anomaly detection. The 3CS mission is considerably less complex than EO-1 but still represents an important step in the integration and flight of onboard autonomy software.

More recent work from NASA Ames Research Center is focused on building the IDEA planning and execution architecture [9]. In IDEA, the planner and execution software are combined into a "reactive planner" and operate using the same domain model. A single planning and execution model can simplify validation, which is a difficult problem for autonomous systems. For EO-1, the CASPER planner and SCL executive use separate models. While this has the advantage of the flexibility of both procedural and declarative representations, a single model would be easier to validate. We have designed the CASPER modeling language to be used by domain experts, thus not requiring planning experts. Our use of SCL is similar to the "plan runner" in IDEA but SCL encodes more intelligence. The EO-1 science analysis software is defined as one of the "controlling systems" in IDEA. In the IDEA architecture, a communications wrapper is used to send messages between the agents, similar to the software bus in EO-1. In the description of IDEA there is no information about the deployment of IDEA to any domains, so a comparison of the performance or capabilities is not possible at this time. In many ways IDEA represents a more AI-centric architecture with declarative modeling at its core and ASE represents more of an evolutionary engineered solution.

ASE on EO-1 demonstrates an integrated autonomous mission using onboard science analysis, replanning, and robust execution. The ASE performs intelligent science data selection that will lead to a reduction in data downlink. In addition, the ASE will increase science return through autonomous retargeting. Demonstration of these capabilities onboard EO-1 will enable radically different missions with significant onboard decision-making leading to novel science opportunities. The paradigm shift toward highly autonomous spacecraft will enable future NASA missions to achieve significantly greater science returns with reduced risk and reduced operations cost.

## References

1. S. Chien, B. Engelhardt, R. Knight, G. Rabideau, R. Sherwood, E. Hansen, A. Ortiviz, C. Wilklow, S. Wichman , "Onboard Autonomy on the Three Corner Sat Mission," Proc i-SAIRAS 2001, Montreal, Canada, June 2001.
2. S. Chien, R. Knight, A. Stechert, R. Sherwood, and G. Rabideau, "Using Iterative Repair to Improve Responsiveness of Planning and Scheduling," Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling, Breckenridge, CO, April 2000. (also casper.jpl.nasa.gov)
3. A.G. Davies, R. Greeley, K. Williams, V. Baker, J. Dohm, M. Burl, E. Mjolsness, R. Castano, T. Stough, J. Roden, S. Chien, R. Sherwood, "ASC Science Report," August 2001. (downloadable from ase.jpl.nasa.gov)
4. Davies, A. G., E.D. Mjolsness, A.G. Gray, T.F. Mann, R. Castano, T.A. Estlin and R.S. Saunders (1999) Hypothesis-driven active data analysis of geological phenomena using semi-autonomous rovers: exploring simulations of Martian hydrothermal deposits. EOS, Trans. Amer. Geophys. Union, 80, no. 17, S210.
5. E. Gat et al., Three-Layer Architectures. in D. Kortenkamp et al. eds. AI and Mobile Robots. AAAI Press, 1998.
6. Goddard Space Flight Center, EO-1 Mission page: http://EO-1.gsfc.nasa.gov
7. Interface and Control Systems, SCL Home Page, sclrules.com
8. M. C. Malin and K. S. Edgett, "Evidence for recent groundwater seepage and surface runoff on Mars," Science, 288, 2330-2335, 2000.
9. N. Muscettola, G. Dorais, C. Fry, R. Levinson, and C. Plaunt, "IDEA: Planning at the Core of Autonomous Reactive Agents," Proceedings of the Workshops at the AIPS-2002 Conference, Tolouse, France, April 2002.
10. NASA Ames, Remote Agent Experiment Home Page, http://ic.arc.nasa.gov/projects/remote-agent/. See also Remote Agent: To Boldly Go Where No AI System Has Gone Before.

Nicola Muscettola, P. Pandurang Nayak, Barney Pell, and Brian Williams. *Artificial Intelligence* 103(1-2):5-48, August 1998

11. The PROBA Onboard Autonomy Platform, http://www.estec.esa.nl/proba/

12. G. Rabideau, R. Knight, S. Chien, A. Fukunaga, A. Govindjee, "Iterative Repair Planning for Spacecraft Operations in the ASPEN System," Intl Symp Artificial Int Robotics & Automation in Space, Noordwijk, The Netherlands, June 1999.

13. S. Chien, B. Cichy, S. Schaffer, D. Tran, G. Rabideau, R. Bote, Dan Mandl, S. Frye, S. Shulman, J. Van Gaasbeck, D. Boyer, Validating the EO-1 Autonomous Science Agent, Working notes of the Workshop on Safe Agents, AAMAS-2003.

14. S. Chien, et al., "The Techsat-21 Autonomous Space Science Agent," International Conference on Autonomous Agents and Multi-agent Systems (AAMAS 2002). Bologna, Italy. July 2002

## Acknowledgement

Portions of this work were performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. We would like to acknowledge the important contributions of Nghia Tang and Michael Burl of JPL, Dan Mandl, Stuart Frye, Seth Shulman, and Stephen Ungar of GSFC, Jerry Hengemihle and Bruce Trout of Microtel LLC, Jeff D'Agostino of the Hammers Corp., Robert Bote of Honeywell Corp., Jim Van Gaasbeck and Darrell Boyer of ICS, Michael Griffin and Hsiao-hua Burke of MIT Lincoln Labs, Ronald Greeley, Thomas Doggett, and Kevin Williams of ASU, and Victor Baker and James Dohm of University of Arizona.